论 文

支持公平服务的 CICQ 分层混合调度策略

扈红超*, 郭云飞, 伊鹏, 兰巨龙

国家数字交换系统工程技术研究中心, 郑州 450002

* 通信作者. E-mail: huhongchao@gmail.com

收稿日期: 2010-10-27; 接受日期: 2011-01-14

国家自然科学基金 (批准号: 60803153)、国家重点基础研究发展计划 (批准号: 2007CB307102) 和国家高技术研究发展计划 (批准号: 2008AA01A323, 2008AA01Z214, 2007AA01Z218) 资助项目

摘要 为到达业务提供性能保障是衡量一个交换系统性能的重要参考. 针对现有联合输入交叉点排队交换结构 (CICQ) 调度策略缺乏基于流的服务质量保障, 探讨了在 CICQ 交换结构实施基于“流”调度的可能性, 提出了一种能够为到达业务流的提供公平服务的分层混合调度策略 (HSFS). HSFS 采用分层的混合调度机制, 每个输入、输出端口可独立地进行变长分组交换, 其复杂度为 $O(1)$, 具有良好可扩展特性. 理论分析结果表明, HSFS 无需加速便能为到达业务提供时延上限、速率和公平性保障. 最后, 基于 SPES 对 HSFS 的性能进行了评估.

关键词 带缓存交叉开关 调度策略 公平服务 分层混合 交换结构

1 引言

当前, 基于联合输入交叉节点排队 (CICQ, combined input-crosspoint queuing) 交换结构^[1]构建的调度机制主要集中在提供高吞吐量^[2~7]、模拟 OQ^[8~10] 和性能保障^[11~13] 3 个方面. 在提供高吞吐量的研究方面, 当输入端口 i 、输出端口 j 的业务流到达率 $\lambda_{ij} \leq 1/N$ 时, LQF-RR 在无需加速的条件下即可获得 100% 的吞吐量^[5]. Giaccone 等^[7] 证明了当交叉点缓存的容量 $B > N$ 时, 仅需 $\lceil 2B/(2B - N) \rceil$ 的加速便可为任意到达的“容许”业务提供 100% 的吞吐量. 异步变长分组调度策略 (LIPS, localized independent packet scheduling)^[6] 基于 round robin 机制, 具有 $O(1)$ 的复杂度, 仅需在每个交叉点维护最大分组长度 $L_{\max}$ 的缓存容量便实现变长分组交换, 且能够在两倍加速条件下提供 100% 的吞吐量. Chang 等基于 CICQ 结构提出了一种基于“帧”大小动态调整的交换机制, 并证明了通过在每个交叉点维护两个信元的缓冲区便可为类似 Poisson 到达的单播流量提供 100% 的吞吐量. 最近, Shen 等基于 Hamiltonian 行走提出了一种能够为到达的业务提供 100% 吞吐量调度算法 (SQUID).

在 OQ 结构的模拟方面, 在 2 倍加速条件下 CICQ 可模拟支持 k 个优先级的先入先出 (first in first out) 输出排队交换结构 (FCFS-OQ)^[8]. 在 3 倍加速条件下 CICQ 可模拟任意调度算法的 OQ 交换结构^[9]. LOOFA (least occupied output first algorithm) 算法在 2 倍加速条件下能够保持持续工作, 并能够在 Push-In, First-Out 排队机制下模拟 OQ 交换结构^[10]. 在性能保障的研究方面, AMFS (adaptive max-min fair scheduling)^[11] 能够在不加速的条件下提供 max-min 公平性. He 等提出一种基

引用格式: 扈红超, 郭云飞, 伊鹏, 等. 支持公平服务的 CICQ 分层混合调度策略. 中国科学: 信息科学, 2012, 42: 410–422

图 1 HSFS 交换系统结构

Figure 1 HSFS switch architecture

于速率的平滑交换结构 (sBUX, smoothed buffered crossbar)^[12], sBUX 仅需在每个交叉点维护两个信元大小的缓存容量便可为实时到达业务提供 100% 的吞吐量, 并能够为每条流提供几近理想的平滑度. 变长分组调度策略 FLAPS (fair and localized asynchronous packet scheduling)^[13] 能够为不同输入、输出端口的流提供性能保障. FLAPS 每个输入和输出单元仅需本地队列信息进行调度. 当交叉点缓存容量为 $4L_{\max}$ 时, 无需加速 FLAPS 算法就可避免缓存溢出, 并可以获得强稳定性 (strong stability) 和时延上限保障, 然而, FLAPS 算法仅能够为定义在端口对上的“流”提供公平服务, 无法对属于同一端口的流进行细分.

针对当前 CICQ 调度机制在业务“流”性能保障方面的不足, 本文提出一种分层的混合调度机制 (HSFS, hybrid and stratified fair scheduling). HSFS 为变长分组调度机制, 每个输入、输出端口可独立地进行变长分组交换, 其复杂度为 $O(1)$, 具有良好可扩展特性. 具体而言, HSFS 将到达交换系统的分组进行分组, 并采用分层的混合调度机制: 组间采用基于时间戳 (timestamp) 的公平调度机制, 而组内采用基于轮询 (round robin) 的调度机制. HSFS 无需加速便能为到达业务提供时延上限、速率和公平性保障. 本文剩余章节安排如下: 第 2 节详细阐述了 HSFS 交换机制; 第 3 节对 HSFS 的性能进行了理论分析; 第 4 节对 HSFS 算法的性能进行仿真评估; 第 5 节是结论, 在总结全文的基础上给出了下一步的研究思考.

2 分层混合调度策略

2.1 系统模型

图 1 给出了一 $N \times N$ 规模的 HSFS 交换系统总体结构, 为避免队头阻塞 (HOLB, head of line blocking), 采用虚拟输出排队 (VOQ, virtual output queuing) 机制缓存分组, 其中 VOQ_{ijk} 缓存到达输入端口 i 、去往输出端口 j 的第 k 条业务流 f_{ijk} ; 交叉点队列 XB_{ij} 缓存到达输入 i 、去往输出 j 的分

表 1 相关符号定义

Table 1 Symbols and definitions

R, N, M	port bandwidth, the switch size and the number of flows
$f_{ijk}(t)$	the k th flow arriving at input i and destined for output j
$\text{VOQ}_{ijk}(t)$	virtual output queue buffers the traffic flow $f_{ijk}(t)$
$r_{ijk}(t)$	required bandwidth of flow f_{ijk} at time t
$w_{ijk}(t)$	scheduling weight of flow f_{ijk} at time t
$q_{ijk}(t)$	queue length of VOQ_{ijk} at time t
G	grouping strategy of HSFS
G_{ij}^h	h^{th} flow group arriving at input i and destined for output j
$\mathbf{G}_{ij}$	flow group set arriving at input i and destined for output j
$\mathbf{B}_{ij}^h$	$\mathbf{B}_{ij}^h = \{f_{ijk} : f_{ijk} \in \mathbf{G}_{ij}^h, q_{ijk}(t) > 0\}$
$r_{ij}^h(t), w_{ij}^h(t)$	bandwidth requirement and scheduling weight of $\mathbf{B}_{ij}^h$
$r_{ij}(t), w_{ij}(t)$	$r_{ij}(t) = \sum_{h=1}^H r_{ij}^h(t), w_{ij}(t) = \sum_{h=1}^H w_{ij}^h(t)$
$\mathbf{F}_{ij}$	flow set arriving at input i and destined for output j
$\mathbf{F}_i$	flow set arriving at input j
$p_{ijh}^{\text{HOL}}, p_{ij}^{\text{HOL}}$	Group $\mathbf{G}_{ij}^h$, the HOL packet of flow set $\mathbf{F}_{ij}$
$V(t), V'(t)$	system virtual time function
$l_{ijh}^{\text{HOL}}, l_{ij}^{\text{HOL}}$	length of $p_{ijh}^{\text{HOL}}, p_{ij}^{\text{HOL}}$
$IS_{ijh}^{\text{HOL}}(t), IF_{ijh}^{\text{HOL}}(t)$	virtual start and finish time of packet p_{ijh}^{HOL}
$IS_{ij}^{\text{HOL}}(t), IF_{ij}^{\text{HOL}}(t)$	virtual start and finish time of packet p_{ij}^{HOL}
$OS_{ij}^{\text{HOL}}(t), OF_{ij}^{\text{HOL}}(t)$	virtual start and finish time of output scheduling
$L_{\max}$	maximum packet length
SDRR_{ijh}	h th intra group scheduler at the input i
IterGS_{ij}	j th inter group scheduler corresponding at the input i
PGS_i	port scheduler corresponding to input i
TOS_j	port scheduler corresponding to output j
$A_{ijk}(t_1, t_2)$	amount of traffic arrived during $[t_1, t_2]$ of flow f_{ijk}

组, 不再为每条流 f_{ijk} 维护独立的虚拟交叉点队列. 到达交换系统的流分组首先进入虚拟输出队列 VOQ_{ijk} , 经输入调度后进入交叉点缓存队列 XB_{ij} , 最后由输出调度后发送到外部链路. 由于交叉点队列的引入, 每个输入、输出端口可独立地执行异步变长 (asynchronous and variable) 分组交换.

为便于描述, 首先给出表 1 描述的相关符号定义. 此外, 为避免网络中传输的数据“分组”和 HSFS 对流进行“分组”概念的混淆, 在下面的描述中, 称数据“分组”为“包”.

2.2 HSFS 算法描述

HSFS 由输入调度器和输出调度器组成, 其中输入调度器由分组策略 (GS, grouping strategy)、组间调度 (InterGS, inter-group scheduling)、组内调度 (IntraGS, intra-group scheduling) 和端口级调度 (PGS, port group scheduling) 组成, 具体而言, 首先根据每条流的权重 w_{ijk} 将 $\mathbf{F}_{ij}$ 划分为 H 个组 $\mathbf{G}_{ij} = \{\mathbf{G}_{ij}^1, \mathbf{G}_{ij}^2, \dots, \mathbf{G}_{ij}^H\}$; 其后, 基于 $\{\mathbf{F}_{ij} : 1 \leq j \leq N\}$ 和组 $\mathbf{G}_{ij} = \{\mathbf{G}_{ij}^1, \mathbf{G}_{ij}^2, \dots, \mathbf{G}_{ij}^H\}$ 进行分层调度:

图 2 HSFS 调度层次结构

Figure 2 Stratified scheduling architecture of HSFS

PGS 采用基于时间戳的调度机制决定各 $\{F_{ij} : 1 \leq j \leq N\}$ 调度顺序; InterGS 采用基于时间戳的调度机制决定各 $\{G_{ij}^1, G_{ij}^2, \dots, G_{ij}^H\}$ 调度顺序; 而 IntraGS 基于轮询机制决定组 G_{ij}^h 内各数据包的调度顺序. 输入调度层次化调度机制组织结构如图 2 所示. 输出调度采用基于时间戳的调度机制 (TOS, time-stamp based output scheduling).

2.2.1 分组策略 (GS)

考察到达任意输入端口 i' 的业务流集合 $F_{i'} = \{f_{ijk} : i = i'\}$, 则去往输出端口 j' 的业务流集合为 $F_{i'j'} = \{f_{ijk} : i = i', j = j'\}$. 分组策略 GS 将所有 $\{F_{ij} : 1 \leq j \leq N\}$ 分别进行分组, 因而共需要 N 个分组单元. 其中, 每个分组策略 $GS_{i'j'}$ 独立地对流集合 $F_{i'j'}$ 进行分组. 为不失一般性, 以到达任意输入端口 i 、去往输出端口 j 的业务流集合 F_{ij} 为例描述 HSFS 算法的分组策略的分组过程. HSFS 分组策略用 $G(\cdot)$ 标识, 分组结果集合 $G_{ij} = \{G_{ij}^1, G_{ij}^2, \dots, G_{ij}^H\}$, 则分组策略 $G(\cdot)$ 描述为 $G_{ij} = G(F_{ij})$.

$G(\cdot)$ 根据流权重 w_{ijk} 将 w_{ijk} 相近的流聚类为一组, 流组 $G_{ij}^h (h \geq 1)$ 分组策略为

$$G_{ij}^h = \left\{ f_{ijk} \in F_{ij} : \frac{1}{B^h} \leq w_{ijk} < \frac{1}{B^{h-1}} \right\}, \quad (1)$$

其中, B 为常量. 令 $r_{\max} = \max_k \{r_{ijk}\}$, $r_{\min} = \min_k \{r_{ijk}\}$, 则 $\frac{1}{B} \leq \frac{r_{\max}}{R}$, 即: $B \geq \frac{R}{r_{\max}}$. 又 $\frac{1}{B^H} \leq \frac{r_{\min}}{R} < \frac{1}{B^{H-1}}$, $\log_B \frac{R}{r_{\min}} \leq H < \log_B \frac{R}{r_{\min}} + 1$, 即 $H = \lceil \log_B \frac{R}{r_{\min}} \rceil$. 取 $B = \frac{R}{r_{\max}}$, 则

$$H = \left\lceil \log_{\frac{R}{r_{\max}}} \frac{R}{r_{\min}} \right\rceil = \left\lceil \frac{\lg R - \lg r_{\min}}{\lg R - \lg r_{\max}} \right\rceil. \quad (2)$$

因而, H 仅取决于链路带宽 R 、最小带宽需求 $r_{\min}$ 和最大带宽需求 $r_{\max}$. 以 $r_{\min} = 1$ bps, $r_{\max} = 1$ Gbps, $R = 160$ Gbps 为例, $H' = \lceil \frac{\lg 160 \times 10^9 - \lg r_{\min}}{\lg 160 \times 10^9 - \lg r_{\max}} \rceil = 6$; 若 $r_{\max} = 10$ Gbps, $H'' = 10$. 可见, H 和交换结构的规模 N 、以及流数 K 无关, 给定 B 、 $r_{\min}$ 和 $r_{\max}$, 分组数 H 为较小常量.

2.2.2 组间调度 (InterGS)

InterGS 根据 F_{ij} 的分组结果在 $G_{ij} = \{G_{ij}^1, G_{ij}^2, \dots, G_{ij}^H\}$ 中选择一个组 G_{ij}^h 作为包的输出组, 因而每个输入端口 i 共需 N 个组间调度策略, 流集合 F_{ij} 的组间调度策略用 InterGS_{ij} 表示. 考察 InterGS_{ij} 的调度过程. 公平调度策略 WFQ^[14]、WF²Q^[15] 通过引入虚拟时间 $V(t)$ 模拟对应的 GPS^[16] 系统过程, 为每条流 f_k 的每个包 p_k^n 计算一个虚拟开始时间标签 S_k^n 和虚拟完成时间标签 F_k^n , 每次

调度时, 选择队头包具有最小虚拟完成时间的流, 即 $k: \min F_k^n(t)$. 若带宽需求 r_k 在调度过程中恒定, 则每个包 p_k^n 的虚拟开始 S_k^n 和结束时间 F_k^n 一旦计算便在系统调度过程中保持恒定, 而和将来包到达无关.

当采用分层调度策略时, 包的到达/离去会导致流的短时“积压”或“空闲”, 引起流所在组带宽需求的变化, 进而引发流所在组和其他组包完成时间的变化, 包虚拟开始和结束时间的不再保持恒定, 而和将来分组的到达相关, 因而必须对分组的到达/离去进行有效地跟踪.

定义 1 积压队列 (backlogged queue) 称 t 时刻队列 VOQ_{ijk} 为积压队列, 若 $q_{ijk}(t) > 0$, 并称流 f_{ijk} 在 t 时刻为积压流, 则 t 时刻 $\mathbf{G}_{ij}^h$ 积压流集合 $\mathbf{B}_{ij}^h = \{f_{ijk} \in \mathbf{G}_{ij}^h : q_{ijk}(t) > 0\}$.

定义 2 事件 (event) 称分组到达/离去调度器 s 为一次事件, 用 e 表示, 第 m 次事件 e_m 发生的时刻用 t_m 表示, 可见在 $[t_m, t_{m+1})$ 的时间间隔内组 $\mathbf{G}_{ij}^h$ 的积压流集合 $\mathbf{B}_{ij}^h$ 无变化.

作为 PGS 和 IntraGS 调度的中间层, InterGS_{ij} 需要选择合适的虚拟时间函数 $V(\cdot)$ 、分组虚拟开始时间戳 $S(\cdot)$ 和分组虚拟结束时间戳 $F(\cdot)$ 的计算方法从而将 PGS 调度层的公平性传递到各 $\mathbf{G}_{ij}^h$, 并确保各 $\mathbf{G}_{ij}^h$ 间的公平性. 令 $r_{ij}^h(t)$ 和 $w_{ij}^h(t)$ 为 t 时刻 $\mathbf{B}_{ij}^h$ 的需求带宽和调度权重, $r_{ij}(t) = \sum_{h=1}^H r_{ij}^h(t)$ 为到达输入 i 、去往输出 j 业务流的汇聚带宽需求, 到达输入端口 i 、去往输出端口 j 的业务流权重为 $w_{ij}(t) = \sum_{h=1}^H w_{ij}^h(t)$, 则时刻 $t \in [t_m, t_{m+1})$ 分组的虚拟时间 $V(t)$ 计算为

$$V(t) = \begin{cases} 0, & \text{if } t = 0; \\ V(t_m) + \tau, & \text{if } w_{ij}(t) = 0; \\ \max\left(V(t_m) + \frac{\tau}{w_{ij}(t)}, \min_h IS_{ijh}^{\text{HOL}}\right), & \text{if } w_{ij}(t) > 0, \end{cases} \quad (3)$$

其中, $\tau = t - t_m$, IS_{ijh}^{HOL} 为组 $\mathbf{B}_{ij}^h$ 队首包的虚拟开始时间. 可见 $V(t)$ 是左连续函数, 取 t_m 时刻值 $V(t_m) = \lim_{t \rightarrow t_m} V(t)$. 记时刻 $t \in [t_m, t_{m+1})$ 组 $\mathbf{B}_{ij}^h$ 的队头包 p_{ijh}^{HOL} 的长度为 l_{ijh}^{HOL} , 则队头包 p_{ijh}^{HOL} 虚拟开始时间戳 $IS_{ijh}^{\text{HOL}}(t)$ 和完成时间戳 $IF_{ijh}^{\text{HOL}}(t)$ 的迭代计算过程为

$$IS_{ijh}^{\text{HOL}}(t) = \begin{cases} \max\{IF_{ijh}^{\text{HOL}}(t), V(A_{ijh}^{\text{HOL}})\}, & \text{if } q_{ij}^h(A_{ijh}^{\text{HOL}-}) = 0; \\ IF_{ijh}^{\text{HOL}}(t), & \text{if } q_{ij}^h(A_{ijh}^{\text{HOL}-}) \neq 0, \end{cases}$$

$$IF_{ijh}^{\text{HOL}}(t) = IS_{ijh}^{\text{HOL}}(t) + \frac{l_{ijh}^{\text{HOL}}}{r_{ij}^h(t)}, \quad (4)$$

其中, 参数 $q_{ij}^h(t)$ 为时刻 t 组 $\mathbf{B}_{ij}^h$ 虚拟输出队列 VOQ_{ij}^h 的长度, $q_{ij}^h(t^-)$ 表示在 t 前一时刻 VOQ_{ij}^h 的长度, A_{ijh}^{HOL} 为 p_{ijh}^{HOL} 的到达时刻. 事实上, VOQ_{ij}^h 是逻辑虚拟输出队列, 存放组 $\mathbf{B}_{ij}^h$ 的队头包的相关信息 (如 p_{ijh}^{HOL} 、 IS_{ijh}^{HOL} 和 IF_{ijh}^{HOL}), 而实际的包仍然存储在流队列 VOQ_{ijk} 中. 当事件 e 发生时, InterGS_{ij} 需重新计算虚拟时间 $V(t)$, 该复杂度为 $O(e)$; 重新计算各组队头包的虚拟开始时间 $IS_{ijh}^{\text{HOL}}(t)$ 和虚拟完成时间 $IF_{ijh}^{\text{HOL}}(t)$ 复杂度为 $O(H)$. 每次分组调度时, InterGS_{ij} 从所有积压组集合 $\mathbf{B}_{ij} = \{\mathbf{B}_{ij}^1, \mathbf{B}_{ij}^2, \dots, \mathbf{B}_{ij}^H\}$ 中选择其队头包 $IS_{ijh}^{\text{HOL}}(t)$ 小于等于 $V(t)$ 且具有最小虚拟完成时间 $IF_{ijh}^{\text{HOL}}(t)$ 的那个组, 即

$$h: \min_h \{IF_{ijh}^{\text{HOL}}(t)\} \text{ s.t. } IS_{ijh}^{\text{HOL}}(t) \leq V(t). \quad (5)$$

从公式 (8) 中可以看出, 对于任意的组 $\mathbf{G}_{ij}^h (1 \leq h \leq H)$ 最坏情况下需要从 H 个队头包 $\{p_{ijh}^{\text{HOL}}\}$ 中选择一个满足 $IS_{ijh}^{\text{HOL}}(t) \leq V(t)$ 且具有最小虚拟完成时间的包作为输出结果, 该复杂度为 $O(\log H)$. 从 2.2.1 小节可知, H 通常为较小的常量值, 因而, H 给定时 $\log H$ 为较小的常量值.

2.2.3 端口调度 (PGS)

PGS 用于决定 $\mathbf{F}_i = \{\mathbf{F}_{ij} : 1 \leq j \leq N\}$ 各流集合 $\mathbf{F}_{ij}$ 的调度顺序, 选择某一集合 $\mathbf{F}_{i'j'}$ 作为数据包的输出集合, 任意的集合 $\mathbf{F}_{ij}$ 可视为组集合 $\mathbf{G}_{ij} = \{\mathbf{G}_{ij}^1, \mathbf{G}_{ij}^2, \dots, \mathbf{G}_{ij}^H\}$ 的汇聚. 同 InterGS, PGS 采用基于时间戳的调度机制确保集合间调度严格的公平性. 与 InterGS 不同之处在于, PGS 的集合数由交换结构的规模 N 确定, 当交换结构规模一定时, $\mathbf{F}_{ij}$ 的数目 $|\mathbf{F}_{ij}| = N$ 为常量. 考察任意输入端口 i , 该端口的端口调度策略用 PGS_i 表示, 记 $r_{ij}(t)$ 为 t 时刻 $\mathbf{F}_{ij}$ 的带宽需求, 则 $r_{ij}(t) = \sum_{h=1}^H r_{ij}^h(t)$, 记 $r_i(t) = \sum_{j=1}^N r_{ij}(t)$ 为输入端口 i 汇聚带宽需求, 则时刻 $t \in [t_m, t_{m+1})$ 虚拟时间 $V'(t)$ 定义为

$$V'(t) = \begin{cases} 0, & \text{if } t = 0; \\ V'(t_m) + \tau, & \text{if } w_i(t) = 0; \\ \max\left(V'(t_m) + \frac{\tau}{w_i(t)}, \min_j IS_{ij}^{\text{HOL}}\right), & \text{if } w_i(t) > 0, \end{cases} \quad (6)$$

其中, t_m 为第 m 次流集合 $\mathbf{F}_i$ 的汇聚带宽需求 $r_i(t)$ 的变化时刻, 且 $\tau = t - t_m$. 由于 $r_i(t)$ 为各流集合 $r_{ij}(t)$ 带宽需求的汇聚, $r_{ij}(t)$ 为各 $r_{ij}^h(t)$ 的汇聚, 因而 $r_i(t)$ 发生在 $r_{ij}^h(t)$ 改变时.

假设 p_{ij}^{HOL} 为时刻 $t \in [t_m, t_{m+1})$ 流集合 $\mathbf{F}_{ij}$ 的队头包, 其长度为 l_{ij}^{HOL} , 则 p_{ij}^{HOL} 的虚拟开始服务时间戳 IS_{ij}^{HOL} 和虚拟完成时间戳 IF_{ij}^{HOL} 的迭代计算过程见公式 (10):

$$IS_{ij}^{\text{HOL}}(t) = \begin{cases} \max\{IF_{ij}^{\text{HOL}}(t), V'(A_{ij}^{\text{HOL}})\}, & \text{if } q_{ij}(A_{ij}^{\text{HOL}-}) = 0, \\ IF_{ij}^{\text{HOL}}(t), & \text{if } q_{ij}(A_{ij}^{\text{HOL}-}) \neq 0, \end{cases}$$

$$IF_{ij}^{\text{HOL}}(t) = IS_{ij}^{\text{HOL}}(t) + \frac{l_{ij}^{\text{HOL}}}{r_{ij}(t)}, \quad (7)$$

其中, $q_{ij}(t)$ 为时刻 t 集合 $\mathbf{F}_{ij}$ 虚拟队列 VOQ_{ij} 的队长, $q_{ij}(A_{ij}^{\text{HOL}-})$ 表示 A_{ij}^{HOL} 前一时刻 VOQ_{ij} 的长度. 同样, 每个 VOQ_{ij} 是一逻辑虚拟队列, 仅存放集合 $\mathbf{F}_{ij}$ 队头包的相关信息 (如 l_{ij}^{HOL} 、 IS_{ij}^{HOL} 和 IF_{ij}^{HOL}), 而实际的包仍存储在其流队列 VOQ_{ijk} 中. 每次速率改变时, PGS_i 重新计算虚拟时间 $V'(t)$, 该复杂度为 $O(e)$; 重新计算各集合队头包的虚拟开始时间 $IS_{ij}^{\text{HOL}}(t)$ 和虚拟完成时间 $IF_{ij}^{\text{HOL}}(t)$ 复杂度为 $O(N)$. 记 $\{\mathbf{F}_{ij} : 1 \leq j \leq N\}$ 中各 $\mathbf{F}_{ij}$ 队头包组成的集合 $\mathbf{P}_i^{\text{HOL}} = \{p_{ij}^{\text{HOL}} : 1 \leq j \leq N\}$, PGS_i 选择 $\mathbf{P}_i^{\text{HOL}}$ 中虚拟开始时间 $IS_{ij}^{\text{HOL}}(t)$ 不大于 $V'(t)$ 且具有最小完成时间 $IF_{ij}^{\text{HOL}}(t)$ 的包所在集合, 即

$$j : \min_j \{IF_{ij}^{\text{HOL}}(t)\} \text{ s.t. } IS_{ij}^{\text{HOL}}(t) \leq V'(t). \quad (8)$$

从公式 (11) 可以看出, PGS_i 最坏情况下需要从 N 个队头包中选择一个满足条件 $IS_{ij}^{\text{HOL}}(t) \leq V'(t)$ 的包作为结果, 当交换结构规模 N 给定时, 其复杂度 $O(\log N)$ 为较小常量值.

记 PGS 调度后选择的包为 p_{ij}^{HOL} , 其实际开始时间戳和结束时间戳分别为 $\widehat{IS}_{ij}^{\text{HOL}}$ 和 $\widehat{IS}_{ij}^{\text{HOL}}$, 则 $\widehat{IF}_{ij}^{\text{HOL}} = \widehat{IS}_{ij}^{\text{HOL}} + l_{ij}^{\text{HOL}}/R$, 随后, p_{ij}^{HOL} 携带虚拟结束时间戳 IF_{ij}^{HOL} 进入交叉点队列 XB_{ij} .

2.2.4 组内调度 (IntraGS)

FRR^[17] 将 DRR^[18] 的一次遍历的所有包组装成“帧”, 获得了短时 WFI(worst-case fairness index)^[15] 公平性. 然而 FRR 需在帧边界重组下一个帧, 带来两方面的问题: 1) 帧边界重组“帧”的复杂度过高 ($O(K)$, 其中 K 为流数目); 2) 属于同一条流 f_k 的分组背靠背 (back-to-back) 进行传输, 最坏情况下分组传输间隔为 $O(L)$, 其中 L 为帧长, 具有较大突发度. 本文采用平滑的 DRR (SDRR, smoothed

DRR) 解决 FRR 的两个问题. 考察组 B_{ij}^h , SDRR 为每条流 $f_{ijk} \in B_{ij}^h$ 按照其权重分配轮询调度份额 s_{ijk} , 即

$$s_{ijk} = B^h w_{ijk} L_{\max}. \quad (9)$$

令 $r_{\max}^h = \max_k \{f_{ijk} \in B_{ij}^h\}$, $r_{\min}^h = \min_k \{f_{ijk} \in B_{ij}^h\}$, 则 $\frac{1}{B^h} \leq \frac{r_{\min}^h}{r_{ij}} \leq \frac{r_{\max}^h}{r_{ij}} < \frac{1}{B^{h-1}}$. 又 $\frac{1}{B^h} = \frac{1}{B} \cdot \frac{1}{B^{h-1}} > \frac{1}{B} \cdot \frac{r_{\max}^h}{r_{ij}}$, 因而,

$$\frac{r_{\max}^h}{r_{\min}^h} \leq B. \quad (10)$$

同 RR/PFQ, SDRR 以“帧”组织轮询调度过程, 如算法 1. 行 (1) 将共享份额计数器 group_deficit 清“0”; 行 (2)~(18) 对至少输出一个包的流组成的链表 active_list 进行轮询: SDRR 首先为流 f_{ijk} 分配份额 s_{ijk} , 更新计数器 $c_{ijk} = c_{ijk} + s_{ijk}$, 并置分配标识位 $s_{ijk} = 1$ (行 (4)~(6)). 若 f_{ijk} 队头包 $p_{ijk}^{\text{HOL}} = \text{HOL}(f_{ijk})$ 长度 l_{ijk}^{HOL} 不大于份额计数器 c_{ijk} , 则将 p_{ijk}^{HOL} 加入到发送链表 fly_list, 更新计数器 $c_{ijk} = c_{ijk} - p_{ijk}^{\text{HOL}}$; 若 $l_{ijk}^{\text{HOL}} > c_{ijk}$, 则将 c_{ijk} 累加到组份额计数器 $\text{group_deficit} = \text{group_deficit} + c_{ijk}$, 将流 f_{ijk} 加入到流轮询链表 active_list' 中. 对 active_list 轮询后, 非空的流 f_{ijk} 转移到 active_list' 中, 进行第 2 轮包的共享份额轮询调度.

行 (19)~(34) 为共享份额轮询调度过程, 其中 active_list' 中存储了队头包长度大于调度份额 c_{ijk} 的流, 即 $\{f_{ijk} : c_{ijk} < l_{ijk}^{\text{HOL}}\}$. 在 group_deficit 大于“0”前, SDRR 依次轮询 active_list': 取出 p_{ijk}^{HOL} , 更新 $\text{group_deficit} = \text{group_deficit} - l_{ijk}^{\text{HOL}}$ 和 $c_{ijk} = c_{ijk} - l_{ijk}^{\text{HOL}}$. 由于 f_{ijk} 预用了一定的份额, 若 p_{ijk}^{HOL} 后一个包 $p_{ijk}^{\text{next}} = \text{next}(p_{ijk}^{\text{HOL}})$ 的长度 l_{ijk}^{next} 满足: $l_{ijk}^{\text{next}} > c_{ijk} + s_{ijk}$, 即下一帧该流无法输出一个包, 则将 f_{ijk} 加入链表 pending_list', 否则加入 pending_list. 当 $\text{group_deficit} \leq 0$ 时, 调整帧发送速率, 将包加入到发送链表 fly_list, 并产生新 active_list 和 pending_list (行 (35)).

2.2.5 输出调度 (TOS)

输出调度 TOS 将包从交叉点缓存队列调度到输出端口, 由于经输入调度对各条流的公平性进行了约束, 因而输出调度仅根据输入调度后分组携带的时间戳进行调度. 由于各输出端口具有相同的调度行为, 这里仅以任意输出 j 为考察输出调度过程, 第 j 个输出调度器记为 TOS_j .

TOS_j 根据交叉点队列集合 $\mathbf{X}_B = \{XB_{ij} : 1 \leq i \leq N\}$ 调度. 记 $\mathbf{P}_j^{\text{HOL}} = \{p_{ij}^{\text{HOL}} : 1 \leq i \leq N\}$ 为 $\mathbf{X}_B$ 的队头包集合, 定义 p_{ij}^{HOL} 的虚拟完成时间戳 OF_{ij}^{HOL} 和虚拟开始时间戳 OS_{ij}^{HOL} 为

$$\begin{aligned} OF_{ij}^{\text{HOL}} &= IF_{ij}^{\text{HOL}} + L_{\max} \left(\frac{H-1}{r_{ij}} + \frac{N-1}{R} \right), \\ OS_{ij}^{\text{HOL}} &= OF_{ij}^{\text{HOL}} - \frac{l_{ij}^{\text{HOL}}}{r_{ij}}, \end{aligned} \quad (11)$$

其中, IF_{ij}^{HOL} 为输入调度后该包携带的虚拟完成时间标签, r_{ijh} 、 r_{ij} 和 R 分别为输入调度阶段包 p_{ij}^{HOL} 所在组 $\mathbf{G}_{ij}^h$ 、集合 $\mathbf{F}_{ij}$ 和输入端口速率, l_{ij}^{HOL} 为包 p_{ij}^{HOL} 的长度. TOS_j 从集合 $\mathbf{P}_j^{\text{HOL}}$ 中选择具有最小虚拟结束时间的包进行调度输出, 即

$$i : \min\{OF_{ij}^{\text{HOL}}\}.$$

Algorithm 1: SDRR scheduler

```

1 group_deficit = 0;
2 begin
3   for each flow  $f_{ijk}$  in active_list do
4     if  $s_{ijk} \neq 1$  then
5        $c_{ijk} += \varsigma_{ijk}, s_{ijk} = 1$ 
6     end
7     if  $f_{ijk} \neq \text{NULL}$  then
8        $p_{ijk}^{\text{HOL}} = \text{HOL}(f_{ijk})$  if  $l_{ijk}^{\text{HOL}} \leq c_{ijk}$  then
9          $c_{ijk} - = l_{ijk}^{\text{HOL}}$ ; remove and add  $p_{ijk}^{\text{HOL}}$  to fly_list; add  $f_{ijk}$  to the tail of active_list;
10      end
11    else
12      group_deficit +=  $c_{ijk}$ ; add  $f_{ijk}$  to the tail of active_list'; break;
13    end
14  end
15  else
16    remove  $f_{ijk}$  from group  $G_{ij}^h$ ;
17  end
18 end
19 for each flow  $f_{ijk}$  in active_list' do
20    $p_{ijk}^{\text{HOL}} = \text{HOL}(f_{ijk})$ ; group_deficit - =  $l_{ijk}^{\text{HOL}}$ ;  $c_{ijk} = c_{ijk} - l_{ijk}^{\text{HOL}}$ ;  $p_{ijk}^{\text{next}} = \text{next}(p_{ijk}^{\text{HOL}})$ 
21   if  $p_{ijk}^{\text{next}} < c_{ijk} + \varsigma_{ijk}$  then
22     add  $f_{ijk}$  to pending_list;
23   end
24   else
25      $c_{ijk} = c_{ijk} + \varsigma_{ijk}$ ; add  $f_{ijk}$  to pending_list';
26   end
27   if group_deficit < 0 then
28     adjust  $G_{ij}^h$ 's transmission rate.
29   end
30   remove and add  $p_{ijk}^{\text{HOL}}$  to the fly_list;
31   if group_deficit  $\leq 0$  then
32     break;
33   end
34 end
35 active_list = pending_list + active_list'; pending_list = pending_list'; pending_list' = active_list' = NULL;
36 end

```

输出调度不再对进入交叉点队列的包细分为流, 由于: 1) 在交叉点不可能为所有的流维护独立的缓存队列, 无法实现基于流的调度; 2) 输入调度对流进行公平性约束, 确保各流进入交叉点的公平性.

用 $\widehat{OS}_{ij}^{\text{HOL}}$ 和 $\widehat{OF}_{ij}^{\text{HOL}}$ 表示输出调度过程中分组实际开始时间戳和结束时间戳, 则

$$\widehat{OF}_{ij}^{\text{HOL}} = \widehat{OS}_{ij}^{\text{HOL}} + l_{ij}^{\text{HOL}}/R.$$

3 性能分析

直观上而言, HSFS 输入调度类似于 $\text{WF}^2\text{Q}^{+ [19]}$ 调度策略, 而输出调度类似于 $\text{PGPS}^{[14]}$ 调度策略, 因而理论上而言, HSFS 应能够为任意的流 f_{ijk} 提供公平性和时延保障. 本节从时延性能和公平性能 (WFI 和 PFI^[20] 公平性) 对 HSFS 的性能进行理论分析, 首先分析时延性能.

3.1 时延性能

记 HSFS 输入调度为 $\mathcal{U} = \langle \text{PS} - \text{GS} - \text{IG} \rangle$, 其中 PS、GS 和 IG 分别表示端口、组间和组内调度策略, $\Omega = \langle \text{IS} - \text{OS} \rangle$ 为输入、输出调度策略对, 例如若输入、输出都采用 GPS, 则 $\Omega = \langle \text{GPS} - \text{GPS} \rangle$, 简记为 $\Omega_{\text{GPS}, \text{GPS}}$. 假设 f_{ijk} 到达过程服从 (σ_{ijk}, r_{ijk}) 漏桶约束, 即

$$A_{ijk}(t_1, t_2) = \int_{t_1}^{t_2} r_{ijk}(t) dt + \sigma_{ijk},$$

其中, $A_{ijk}(t_1, t_2)$ 为 $[t_1, t_2)$ 内 f_{ijk} 的到达流量, σ_{ijk} 为仅和流 f_{ijk} 相关的常量.

定理 1 对于存在 H 个组的集合 $\mathbf{G}$, 若组间调度机制采用 InterGS, 则该调度机制下组 $\mathbf{G}_h$ 第 n 个包的离去时间与对应 GPS 系统中该组第 n 个分组的离去时间满足:

$$\hat{F}_{h, \text{InterGS}}^n(t) - \bar{F}_{h, \text{GPS}}^n(t) \leq (H-1) \frac{L_{\max}}{r},$$

其中, r 为组集合 $\mathbf{G}$ 的输出共享链路带宽.

证明 由于篇幅限制, 这里不再给出证明, 详细证明过程请参考文献 [21].

定理 2 设 $\mathbf{F}_i = \{\mathbf{F}_{ij} : 1 \leq j \leq N\}$ 中集合数为 N , $\mathbf{F}_{ij} = \{\mathbf{G}_{ij}^h : 1 \leq h \leq H\}$ 分组数为 H_j , 若流 $f_{ijk} \in \mathbf{F}_i$ 的到达服从 (r_{ijk}, σ_{ijk}) 的漏桶约束, 则采用 $\mathcal{U} = \langle \text{SDRR} - \text{InterGS} - \text{PGS} \rangle$ 调度机制时集合 $\mathbf{G}_{ij}^h$ 第 n 个包的完成时间 $\widehat{IF}_{ijh, \mathcal{U}}^n(t)$ 与对应 GPS 系统该包完成时间 $\widetilde{IF}_{ijh, \text{GPS}}^n(t)$ 满足关系

$$\widehat{IF}_{ijh, \mathcal{U}}^n(t) - \widetilde{IF}_{ijh, \text{GPS}}^n(t) \leq L_{\max} \left(\frac{H_j - 1}{r_{ij}} + \frac{N - 1}{R} \right),$$

其中, r_{ij} 为集合 $\mathbf{F}_{ij}$ 的汇聚输出速率.

证明 由于篇幅限制, 这里不再给出证明, 详细证明过程请参考文献 [21].

定理 3 若交叉点队列 $\{XB_{ij}\}$ 在 $[t_1, t_2)$ 内处于持续“积压”状态, 那么对应于该队列的第 n 个包的完成时间 $\widehat{OF}_{ij, \text{TOS}}^n$ 和 GPS 下包的完成时间 $\widetilde{OF}_{ij, \text{GPS}}^n$ 满足

$$\widehat{OF}_{ij, \text{TOS}}^n - \widetilde{OF}_{ij, \text{GPS}}^n \leq \frac{L_{\max}}{R}.$$

证明 定理 (3) 的证明方法同定理 (1) 的证明, 这里不再给出证明.

定理 4 若 CICQ 交换系统采用 Ω_{HSFS} 调度策略, 则组 $\mathbf{G}_{ij}^h$ 的第 n 个包 p_{ijk}^n 的实际离去时间 $\hat{F}_{ijh, \Omega_{\text{HSFS}}}^n$ 和采用调度策略 Ω_{GPS} 时该包的离去时间 $\tilde{F}_{ijh, \Omega_{\text{GPS}}}^n$ 存在如下关系:

$$\hat{F}_{ijh, \Omega_{\text{HSFS}}}^n - \tilde{F}_{ijh, \Omega_{\text{GPS}}}^n \leq L_{\max} \left(\frac{H - 1}{r_{ij}} + \frac{N}{R} \right).$$

证明 由于篇幅限制, 这里不再给出证明, 详细证明过程请参考文献 [21].

3.2 公平性能

定理 5 当采用 HSFS 调度策略时 CICQ 系统能够为流 f_{ijk} 提供 WFI 公平性.

证明 由于篇幅限制, 这里不再给出证明, 详细证明过程请参考文献 [21].

定理 6 当采用 HSFS 调度策略时 CICQ 系统能够为流 f_{ijk} 提供 PFI 公平性.

证明 由于篇幅限制, 这里不再给出证明, 详细证明过程请参考文献 [21].

4 仿真实验

本节从时延抖动、带宽分配的公平性和吞吐量 3 个方面对 HSFS 算法的性能进行仿真评估. 实验采用 C++ 语言开发的交换系统性能仿真评估系统 (SPES, switching performance evaluation system)^[22]. SPES 仿真环境配置如下: 交换结构的规模为 16×16 ; 输入、输出端口的带宽归一化为 1; 由于 HSFS 为变长分组交换, 仿真中产生的分组的长度在 $[40, 1500]$ 内均匀变化值, 单位为字节 (bytes); 流量到达模型采用 Markov 调制的 Poisson 过程; 流量分布采用均匀和非均匀两种业务流分布模型, 用 λ_{ij} 表示到达输入 i 、去往输出 j 的速率, 则对于均匀分布: $\lambda_{ij} = \frac{R}{N}$; 对于非均匀分布

$$\lambda_{ij} = \begin{cases} R\left(w + \frac{1-w}{N}\right), & \text{if } i = j; \\ R\frac{1-w}{N}, & \text{if } i \neq j, \end{cases}$$

其中, w 为不均衡指数, 显然 w 越大, 流量分布越集中. 由于 HSFS 为基于流的调度机制, 为便于分析, 对于任意输入端口 i 和输出端口 j , 产生两条流 f_{ij1} 和 f_{ij2} , 且 $r_{ij1} = 3r_{ij2}$. 取分组参数 $B = 2$, 则 f_{ij1} 和 f_{ij2} 分属不同的组 $G_{ij}^1 = \{f_{ij1}\}$ 和 $G_{ij}^2 = \{f_{ij2}\}$, 共有 $2N$ 个流组.

4.1 GPS 相对时延

采用均匀和非均匀两种流量分布模型评估 HSFS 的 GPS 相对时延性能. 为了便于说明问题, 这里仅以流 f_{ij1} 为例分析分组的 GPS 相对时延性能. 图 3(a) 和 (b) 分别给出了均匀和非均匀业务到达下 HSFS 算法的 GPS 平均相对时延 (ΔD_{av}) 性能随流量强度和分布的变化曲线. 为了便于说明问题, 同时给出了理论上 GPS 相对时延的上下限. 可以看出, 在均匀流量到达下, ΔD_{av} 随着流量强度的增长而增长. 在非均匀流量到达下, 相对时延 ΔD_{av} 随着不均衡指数 w 的不断增加而变化, 从最大值逐渐趋近于“0”, 并在 $w = 0$ 时取得了极大值, 这是由于 $w = 0$ 时非均匀分布演化为均匀分布, 各端口之间的影响最大.

4.2 公平性能保障

采用均匀和非均匀两种流量分布模型评估 HSFS 带宽分配的公平性. 带宽公平性通过各条流 f_{ijk} 的预约带宽和获得的实际带宽之间相对比率 ξ 衡量:

$$\xi_i = \frac{W_{ijk}/r_{ijk}}{W_{ijk'}/r_{ijk'}}. \quad (12)$$

图 4(a) 和 (b) 给出了均匀和非均匀业务分布下 HSFS 算法的公平性能仿真结果. 在均匀分布下, 流 f_{111} 和 f_{112} 获的带宽随到达强度的增大而增大, 然而 ξ_1 始终为 1; 在非均匀分布下, 流 f_{111} 、 f_{111} 、 f_{211} 和 f_{212} 获得带宽随 w 变化而变化, 然而相对比率 ξ_2 的值都在 1.0 附近波动.

图 3 均匀和非均匀业务到达下 HSFS 算法的时延性能

Figure 3 Delay performance of HSFS under uniform and non-uniform traffic arrivals. (a) Uniform; (b) non-uniform

图 4 均匀和非均匀业务到达下 HSFS 算法的带宽性能

Figure 4 Bandwidth performance of HSFS under uniform and non-uniform traffic arrivals. (a) Uniform; (b) non-uniform

图 5 吞吐量性能仿真结果

Figure 5 Throughput performance of HSFS under non-uniform traffic arrival

4.3 吞吐量性能

吞吐量性能采用文献 [22] 中的计算方法进行评估, 用 $\lambda_{\max}$ 表示交换系统稳定条件下任意输入端口 i 容许的最大业务流到达速率, 则交换系统的吞吐量 $T = \lambda_{\max}$.

图 5 给出了非均匀分布下 HSFS 算法的吞吐量性能仿真结果. 可以看出, 当不平衡指数 w 从 0.1 增长到 1.0 过程中, HSFS 算法的吞吐量均接近理论上限值 1.0.

5 结束语

本文提出了一种能够为流的提供公平服务的混合调度策 (HSFS, hybrid and stratified fair scheduling). 同现有调度策略不同, HSFS 采用分层混合调度机制, 根据流权重进行分组, 组间采用基于时间戳的调度策略确保包的时延和公平性保障, 组内采用基于轮询调度策略以获得 $O(1)$ 的时间复杂度. 此外, 每个输入、输出可独立地变长分组交换. 理论分析表明, HSFS 无需加速便能为流提供时延上限和公平性保障. 最后, 基于 SPES (switching performance evaluation system) 对 HSFS 的性能进行了评估. 依托于这一思想, 后期将在支持区分服务方面进行深入研究.

参考文献

- 1 Nabeshima M. Performance evaluation of a combined input- and crosspoint-queued switch. *IEICE Trans Commun*, 2000, E83-B: 737–741
- 2 Rojas-Cessa R, Oki E, Jing Z, et al. On the combined input-crosspoint buffered switch with round-robin arbitration. *IEEE Trans Commun*, 2005, 53: 1945–1951
- 3 Mhamdi L, Hamdi M. MCBF: A high-performance scheduling algorithm for buffered crossbar switches. *IEEE Commun Lett*, 2003, 7: 451–453
- 4 Zhang X, Bhuyan L. An efficient scheduling algorithm for combined input-crosspoint-queued (CICQ) switches. In: *Proc. of the IEEE Globecom*. New York: IEEE Communications Society, 2004. 1168–1173
- 5 Javidi T, Magill R, Hrabik T. A high-throughput scheduling algorithm for a buffered crossbar switch fabric. In: *Proceedings of the IEEE Int'l Conf. on Communications (ICC)*. New York: IEEE Communications Society, 2001. 1586–1591
- 6 Pan D, Yang Y. Localized independent packet scheduling for buffered crossbar switches. *IEEE Trans Comput*, 2009, 58: 260–274
- 7 Chang C, Hsu Y, Cheng J, et al. A dynamic frame sizing algorithm for CICQ switches with 100% throughput. In: *Proceedings of IEEE INFOCOM*. New York: IEEE Communications Society, 2009. 738–746
- 8 Magill B, Rohrs C, Stevenson R. Output-queued switch emulation by fabrics with limited memory. *IEEE J Select Areas Commun*, 2003, 21: 606–615
- 9 Chuang S T, Iyer S, McKeown N. Practical algorithms for performance guarantees in buffered crossbars. In: *Proceedings of IEEE INFOCOM*. New York: IEEE Communications Society, 2005. 981–991
- 10 Turner J. Strong performance guarantees for asynchronous crossbar schedulers. *IEEE/ACM Trans Netw*, 2009, 17: 1017–1028
- 11 Zhang X, Mohanty S, Bhuyan L. Adaptive max-min fair scheduling in buffered crossbar switches without speedup. In: *Proceedings of IEEE INFOCOM*. New York: IEEE Communications Society, 2007. 454–462
- 12 He S, Sun S, Guan H, et al. On guaranteed smooth switching for buffered crossbar switches. *IEEE/ACM Trans Netw*, 2008, 16: 718–731
- 13 Pan D, Yang Z, Makki K, et al. Providing performance guarantees for buffered crossbar switches without speedup. In: *Proc ICST QShine*. Berlin: Springer, 2009. 297–314
- 14 Demers A, Keshav S, Shenker S. Analysis and simulation of a fair queuing algorithm. In: *ACM SIGCOMM'89*. New York: ACM Press, 1999. 1–12
- 15 Bennett J C R, Zhang H. WF²Q: Worst-case fair weighted fair queuing. In: *IEEE INFOCOM'99*. New York: IEEE Communications Society, 1996. 120–128
- 16 Parekh A, Gallager R. A generalized processor sharing approach to flow control in integrated services networks: the single node case. *IEEE/ACM Trans Netw*, 1993, 1: 344–357
- 17 Yuan X, Duan Z. FRR: a proportional and worst-case fair round robin scheduler. *IEEE Trans Comput*, 2009, 58: 365–379
- 18 Shreedhar M, Varghese G. Efficient fair queuing using deficit round robin. *IEEE/ACM Trans Netw*, 1996, 4: 375–385

- 19 Bennett J C R, Zhang H. Hierarchical packet fair queueing algorithms. *IEEE/ACM Trans Netw*, 1997, 5: 675–689
- 20 Golestani S. A self-clocked fair queueing scheme for broadband applications. In: *Proceedings of IEEE INFOCOM*, New York: IEEE Communications Society, 1994
- 21 Hu H C, Guo Y F, Yi P. Design and implementation of fair scheduling algorithms in combined input and crosspoint queued switches. Technical Report. National Digital Switching System Engineering & Technological R&D Center. Zhengzhou, 2010
- 22 Hu H C, Yi P, Guo Y F. Design and implementation of high performance simulation platform for switching and scheduling. *J Softw*, 2008, 19: 1036–1050

Achieving fair service with a hybrid scheduling scheme for CICQ switches

HU HongChao*, GUO YunFei, YI Peng & LAN JuLong

Department of Broadband Information Networks, National Digital Switching System Engineering & Technological R&D Center (NDSC), Zhengzhou 450002, China

*E-mail: huhongchao@gmail.com

Abstract Providing performance guarantees for arriving traffic flows has become an important measure for today's routing and switching systems. However, none of current scheduling algorithms built on CICQ (combined input and cross-point buffered) switches can provide flow level performance guarantees. Aiming at meeting this requirement, the feasibility of implementing flow level scheduling is discussed thoroughly. Then, based on the discussion, it comes up with a hybrid and stratified fair scheduling (HSFS) scheme, which is hierarchical and hybrid, for CICQ switches. With HSFS, each input port and output port can schedule variable length packets independently with a complexity of $O(1)$. Theoretical analysis show that HSFS can provide delay bound, service rate and fair performance guarantees without speedup. Finally, we implement HSFS in SPES (switch performance evaluation system) to verify the analytical results.

Keywords switching fabric, scheduling policy, load balancing, dynamical re-routing, switching fabric

HU HongChao was born in 1982. He received M.E. degree in communication and information engineering from National Digital Switching System Engineering & Technological R&D Center (NDSC), Zhengzhou in 2007. He has been doing research in the areas of high performance routing and switching architectures, traffic load-balancing technologies, and traffic management and control technologies in the NDSC. He is currently working towards his Ph.D.

degree at NDSC, and engaged in researching and developing large scale traffic management and control systems. He is a member of the IEICE.